VI & VIC

Nr. 2 - 2. årg. - April-maj 1983 | Pris 10,95 kr.

VIC holder styr på dine bankoplader - eller den ryster posen og råber op

»Hvis man nu kunne lade computeren holde øje med bankopladerne, kunne man klare mange flere«. Sådan lød et hjertesuk fra en af VICtors venner. Han er medlem af en walkie talkie klub, og mindst én gang om ugen går det løs.

- Jeg vil jo gerne støtte foretagendet ved at købe mange plader, og der er god tid til at taste dem ind i VIC på forhånd, fortsætter vores bankoman, som vi naturligvis ikke kunne lade i stikken. Det program, vi her præsenterer, kan holde øje med temmelig mange plader på én gang, men det skal erkendes, at jo flere plader, der kommer til, jo længere tid er programmet om at læse sig igennem alle tallene og kontrollere, om de findes på pladerne. Når et tal er fundet, skal det jo også lige tjekkes efter, om de fire andre tal i samme række har været råbt op - og har de det, er der jo også en mulighed for, at de to andre rækker allerede er hjemme. Men op til en snes plader kan man vist godt påtage sig.

På bånd

For at gøre livet lettere, kan det lade sig gøre at gemme de plader, man én gang har indtastet, på bånd. Som en ekstra funktion kan programmet også fungere som opråber. Tallene fra 1 til 90 genereres i tilfældig rækkefølge, og man kan til enhver tid få en oversigt over, hvilke tal, der allerede er råbt op.

Ved starten af programmet spørges der om, der skal hentes plader fra tape, om der skal indtastes nye plader - eller om programmet skal fungere som »opråber«. TAL står der på skærmen, og det betyder, at man kan få tal fra programmet.

GET$-kommando

Skal der indtastes plader, skal man først svare på, hvor mange plader, det drejer sig om. Det er af betydning, fordi de tabeller, hvor de mange tal skal gemmes, skal dimensioneres med DIM, og det er en kommando, der kun kan



FORTH kan give din VIC nye muligheder

Læs side 8

Professionelle bruger VIC i undervisningen

Læs side 10

To måder at bruge VIC til lektiehjælp

Læs side 12

Fortsat fra forsiden

VIC holder styr på dine bankoplader

```
0 SK=4*(PEEK(36866)AND128)+64*(PEEK(36869)AND120):DIMTA(90):GOTO9
1 POKE36878,15:POKE36876,225:FORI=1TO50:NEXT:POKE36876,0:RETURN
2 GETS$:IFS$=""THEN2
3 RETURN
4 POKE36878,15:POKE36875,188:FORI=1TO150:NEXT:POKE36875,0:RETURN
5 PRINT"▯▮"ST$:PRINT" ▮       NYT SPIL▮":PRINTST$:GOSUB1:B=0
6 FORP=1TOPL:FORR=1TO3:FORT=1TO5:A(P,R,T)=ABS(A(P,R,T)):NEXTT,R,P:S$="":GOSUB7:G
OTO1000
7 FORD=1TO1000:NEXT:RETURN
8 FORV=1TO3:FORS=225TO185STEP-1:POKE36875,S:NEXT:POKE36875,0:NEXT:RETURN
9 DEFFNA(P)=SK+47+2*INT(P/10)+44*(P-INT(P/10)*10)
10 S$="   0 1 2 3 4 5 6 7 8    ┌┬┬┬┬┬┬┬┬┐   0|▮| | | | | | | | |"
11 S$=S$+"    ├┼┼┼┼┼┼┼┼┤   1| | | | | | | | | |
12 S$=S$+"    ├┼┼┼┼┼┼┼┼┤   2| | | | | | | | | |
13 S$=S$+"    ├┼┼┼┼┼┼┼┼┤   3| | | | | | | | | |
14 X$=S$+"    ├┼┼┼┼┼┼┼┼┤   4| | | | | | | | | |
15 K$=K$+"    ├┼┼┼┼┼┼┼┼┤   5| | | | | | | | | |
16 K$=K$+"    ├┼┼┼┼┼┼┼┼┤   6| | | | | | | | | |
17 K$=K$+"    ├┼┼┼┼┼┼┼┼┤   7| | | | | | | | | |
18 K$=K$+"    ├┼┼┼┼┼┼┼┼┤   8| | | | | | | | | |
19 K$=K$+"    ├┼┼┼┼┼┼┼┼┤   9| | | | | | | | | |
20 X1$=K$+"    └┴┴┴┴┴┴┴┴┘"
60 ST$="   ****************"
70 PRINT"▯▮"ST$:PRINT"▮       BANKO-SYS▮":PRINTST$
80 PRINT"▮▮A  PLADER FRA TAPE":PRINT"▮B  PLADER SKAL            INDTASTES":GOSUB1
85 PRINT"▮C  TAL"
90 GETS$:IFS$<"A"ORS$>"C"ORS$=""THEN90
91 IFS$="A"THEN400
92 IFS$="C"THEN10000
100 PRINT"▮▮HVOR MANGE PLADER?":GOSUB1:INPUTPL
110 DIMA(PL,3,5)
210 FORP=1TOPL
220 PRINT"▮▯PLADE NR."P"▮ AF"PL
230 FORR=1TO3:PRINT
240 FORT=1TO5:GOSUB1:PRINT"▮▮▮SKRIV TAL (2 CIFRE)":FORI=1TOR:PRINT"▮▮":NEXT
250 GETA$:IFA$=""ORA$<"0"ORA$>"9"THEN250
251 IFINT(VAL(A$)/2)=VAL(A$)/2THENPRINT"▮":GOTO255
252 PRINT"▪"
255 PRINTTAB(VAL(A$)*2)A$;
260 GETB$:IFB$=""ORB$<"0"ORB$>"9"THEN260
261 IFA$="9"ANDB$<>"0"THEN260
262 IFA$="0"ANDB$="0"THEN260
264 IFA$="0"THENPRINT"▮";
265 PRINTB$"▮";:A(P,R,T)=VAL(A$+B$)
299 NEXTT,R,P
300 PRINT:PRINT"▯▮▮SKAL TALLENE GEMMES?":GOSUB1:GOSUB2
320 IFS$<>"J"THEN1000
330 OPEN1,1,1,"BANKO"
340 PRINT#1,PL
350 FORP=1TOPL:FORR=1TO3:FORT=1TO5
360 PRINT#1,A(P,R,T)
370 NEXTT,R,P
380 CLOSE1:GOTO0300
400 OPEN1,1,0,"BANKO"
440 INPUT#1;PL
450 FORP=1TOPL:FORR=1TO3:FORT=1TO5
460 INPUT#1,A(P,R,T)
470 NEXTT,R,P
480 CLOSE1
1000 PRINT"▮▮▮TRYK ▮TAST▀":GOSUB2
1001 B=0
1002 PRINT"▯"X$;:PRINTX1$:PRINT"
     |  |▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮|";
1003 FORP=1TOPL:FORR=1TO3:FORT=1TO5:IFA(P,R,T)
     <90ANDA(P,R,T)>0THENPOKEFNA(A(P,R,
T)),87
1004 IFA(P,R,T)=90THENPOKESK+503,87
1005 NEXTT,R,P
1010 GOSUB2:IFS$="↑"THEN5
```

VI & VIC

VI & VIC udgives af commodore data a/s,
Bjerrevej 67, 8700 Horsens.
Redaktion og tilrettelæggelse:
Baghuset Grafik/Design, Vejle.
Ansvarshavende redaktør: Steen Venbjerg.
Annoncer: Lise Tylvad,
Æblehaven 7, 7100 Vejle. Tlf. (05) 82 52 97
Tryk: Baghuset Grafik/Design, Vejle.
Redaktion og ekspedition:
Kornager 29, 7100 Vejle.
Giro nr. 6 17 39 93
Kopiering af bladets indhold er ikke tilladt, men særtryk af de enkelte artikler kan leveres.
ISSN 0108-2213

```
1011 IFS$="←"THEN14000
1012 A$=S$:IFA$<"0"ORA$>"9"THENGOSUB4:GOTO1010
1020 PRINTTAB(8)"■■■■"A$" ■";
1030 GOSUB2:B$=S$:IFB$<"0"ORB$>"9"THENGOSUB4:GOTO1030
1040 IFA$="9"ANDB$<>"0"THENGOSUB4:GOTO1030
1050 IFA$="0"ANDB$="0"THENGOSUB4:GOTO1030
1060 PRINTB$"  ";:GOSUB1
1070 TA=VAL(A$+B$):GOSUB2000:IFB=1THEN5
1071 IFRA=1THENRA=0:GOTO1000
1072 IFTA<>90THENPOKEFNA(TA),81
1073 IFTA=90THENPOKESK+503,81
1080 GOTO1010
2000 REM RUTINE DER SØGER RIGTIGE
2010 FORP=1TOPL:FORR=1TO3:FORT=1TO5
2020 IFA(P,R,T)=TATHENA(P,R,T)=-A(P,R,T):GOSUB3000
2030 NEXTT,R,P:RETURN
3000 FORX=1TO5:IFSGN(A(P,R,X))=1THENRETURN
3010 NEXT:GOSUB12000:PRINT
3020 GOSUB4000:RETURN
4000 FORX=1TO3:FORY=1TO5:IFSGN(A(P,X,Y))=1THENRETURN
4010 NEXTY,X:GOSUB13000:RETURN
4030 RETURN
10000 REM TAL OPRAAB
10020 PRINT"■":PRINTST$:PRINT"■          TAL■":PRINTST$
10030 PRINT"■■TRYK TAST FOR NYT TAL"
10035 PRINT"■■TRYK ← FOR KONTROL"
10036 PRINT"■■TRYK ↑ FOR NYT SPIL"
10040 NT=INT(RND(1)*90)+1
10042 IFX=90THEN:GOSUB7:GOSUB15000:S$="↑":POKE198,0:GOSUB1:GOTO10055
10045 IFTA(NT)=1THEN10040
10050 GOSUB2
10051 IFS$="←"THENGOSUB15000
10055 IFS$="↑"THENX=0:GOSUB15000:FORT=1TO90:TA(T)=0:NEXT:GOTO10000
10060 X=X+1:TA(NT)=1:PRINTTAB(10)"┌────┐":PRINT"TAL NR"X"■":"TAB(10)"|"NT;:PRINTT
AB(15)"|
10070 PRINTTAB(10)"└────┘■■":GOSUB1
10080 GOTO10040
11001 FORV=1TO90:IFTA(V)=1THENPRINTV;
11002 NEXT:PRINT:RETURN
11010 PRINT"■TRYK ← FOR AT FORTS.":GOSUB1:INPUT"■TAL";TT$:IFTT$="←"THENPRINT"■":
RETURN
11015 IFASC(TT$)<48ORASC(TT$)>57ORVAL(TT$)>90ORVAL(TT$)<1THENGOSUB4:PRINT"■DUER
IKKE!":GOSUB7:GOTO11010
11020 IFTA(VAL(TT$))=0THENGOSUB4:PRINT"■IKKE TRUKKET":GOSUB7:GOTO11010
11030 PRINT"■OK":GOSUB1:GOSUB7:GOTO11010
12000 REM EN RAEKKE***
12005 RA=1:PRINT"■":PRINTST$:PRINT"■       EN RAEKKE■":PRINT"      PLADE NR."P
12010 PRINT"■"ST$:FORG=1TO5:GOSUB1:NEXT
12020 PRINT"■■":FORT=1TO5:PRINTTAB(T*3-2)ABS(A(P,R,T));:NEXT
12030 RETURN
13000 REM BANKO   ****
13005 B=1:PRINT"■":PRINTST$:PRINT"■          BANKO■":PRINT"      PLADE NR."P
13006 PRINT"■"ST$:GOSUB8
13010 FORR=1TO3:GOSUB12020:NEXT:PRINT:PRINT"■TRYK ■TAST■":GOSUB2:RETURN
14000 REM**KONTROL AF PLADER***
14010 FORP=1TOPL:PRINT"■■PLADE"P:GOSUB13010:PRINT:NEXT:GOTO1002
15000 REM **OVERSIGT OVER TRUKNE TAL***
15010 PRINT"■"X$;:PRINTX1$:PRINT"   TRYK ←          | |";
15020 FORV=1TO89:IFTA(V)=1THENPOKEFNA(V),81
15030 NEXT:IFTA(90)=1THENPOKESK+503,81
15040 GOSUB2:IFS$="←"THENPRINT"■":RETURN
15045 GOTO15040

READY.
```

gives én gang. Under indskrivningen af tallene behøver man ikke at trykke RETURN hver gang et tal er skrevet. Til gengæld er man nødt til at sætte et nul foran de 1-cifrede tal. Denne del af programmet gør brug af GET$-kommandoen, og det betyder også, at der ikke er nogen blinkende markør på skærmen. Til gengæld er der lagt nogle små kontroller ind i programmet, så det ikke er muligt at indtaste ulovlige tal (d.v.s. tal over 90 eller mindre end 1.

Som et lille raffinement bliver tallene automatisk skrevet på skærmen med alle 10'ere under hinanden, 20'ere under hinanden etc. På hver plade skal der skrives 15 tal, og de placeres automa-

tisk med fem tal i tre rækker - man skal blot taste ind nøjagtig som tallene står på den plade, man skal spille på. Programmet tildeler hver enkelt plade et løbenummer, så man kan finde ud af, hvilken plade, det egentlige var, man vandt på.

Gemmes på bånd

Når alle tal er indtastet, spørger programmet, om tallene skal gemmes, svarer man »J«, må man være klar med båndoptageren. Efter at tallene er lagt på bånd, vender programmet tilbage til samme spørgsmål for at give brugeren mulighed for at gemme tallene i flere kopier, hvis man vil være på den sikre side.
Efter at være fædig med at gemme tallene, går programmet direkte videre til spillet - som man også ville komme til, hvis man havde indlæst tal fra et eksisterende bånd. Her viser skærmbillede et 10x9 skema, hvor der er hægtet et ekstra rum på forneden. Her repræsenterer hver søjle 10'erne, mens hver række holder styr på enerne. Alle de tal, der findes på pladerne i programmet plottes automatisk ind i diagrammet, idet tallet 90 har fået sin særlige plads nederst til højre.

Nu kan spillet begynde, og ligesom under indtastningen af tallene behøver man nu kun trykke tallene ind (altid to cifre), og hvor de tal, man har på pladerne, er angivet med en ring, bliver der sat en sort klat i de pladser, der repræsenterer de udtrukne tal.

Opråberen

Den sidste funktion i bankoprogrammet er opråberen. Her trykker man ganske enkelt på en hvilken som helst tast på VIC - og straks meddeler den et tilfældigt valgt tal fra 1 til 90. Og den holder vel at mærke øje med, at ingen tal bliver nævnt to gange.
de fire sidst trukne tal kan ses på skærmen, men når nogen har fyldt en række eller en plade, skal tallene kontrolleres. Det sker med pil-til-venstre tasten, der kalder det samme diagram frem, som bruges under spillet - og samtlige trukne tal plottes ind. Skal man i gang med et nyt spil, bruges også her pil-opad tasten.
Det vil naturligvis være fristende at lave en snirkel i spilprogrammet, så man på samme VIC-20 kan være opråber og deltage i spillet, men det vil vi helt overlade til læserne - for resultatet kan jo hurtigt blive, at VIC sidder og spiller banko med sig selv . . .

Gennemse

Man kan til enhver tid gennemse sine spilleplader ved at trykke på pil-til-venstre tasten. Når man med denne tast har »bladret« sig gennem alle plader, vender man tilbage til diagrammet, men nu er der kun indplottet de tal, der stadig mangler at blive råbt op. Skulle det ulykkelige ske, at en anden spiller får banko, trykker man på pil-opad tasten, hvorefter man er klar til et nyt spil med de samme spilleplader.
Får man selv fyldt en række eller ligefrem banko, giver programmet naturligvis melding om det. Man må sige, det er en svaghed ved programmet, at det FØRST rapporterer om hele rækker - og bagefter om BANKO. I værst tænkelige fald kan konsekvensen blive, at en »manuel« spiller, der får banko i samme stund, løber med gevinsten, fordi han er hurtigere.
Den måde, der er benyttet for at markere, at et tal er råbt op, er en vending af fortegnet. I tabellen gøres de tal negative, som bliver udtrukket, og når alle fem tal i en række er negative, er rækken fyldt. Det ville være fristende at lægge en funktion ind i programmet, der opretter en lille tabel med »kritiske« tal - altså tal, som man véd vil give banko, når de trækkes - men ajourføringen af en sådan tabel vil tage tid - og selv om man vil tjekke først i denne tabel for at se, om der er banko, slipper man ikke for at løbe hele stakken igennem. Resultatet vil altså uvægerligt blive, at programmet kommer til at køre langsommere - og hvad nytter det, man har gevinst på sine plader, hvis alle de andre er gået hjem, når man råber »Banko!«?

Skønhedsfejl i program for check-regnskab er lette at fjerne igen

Det program for check-regnskab, som Commodore udbyder, er behæftet med nogle skønhedsfejl, men heldigvis er det ikke værre, end at det kan løses. Der er en enkelt linie, der skal fjernes, og der skal foretages en rettelse af tre andre linier.
Fra Lars Pedersen, Elmebakken 23 i Snekkersten, har vi modtaget følgende opfordring til at løse mysteriet:

Jeg mener, at det ikke fungerer hensigtsmæssigt skærmmæssigt. Især funktion »F3 Opslag« synes at skabe »rod« på skærmen, såfremt systemet indeholder mere end 4 posteringer og »n« anvendes for at se de 4 forudgående posteringer. Heller ikke »F1 ajourføring« opfører sig regelret, når der indtastes mere end 1 postering i samme ajourføring.
Jeg ville være meget taknemmelig, såfremt VI & VIC har mulighed for at »Verify« programmet.
I øvrigt særdels tilfreds med bladet om det blot kom oftere end hver 2. måned.

Med venlig hilsen
Lars Pedersen

Fra Commodore Data i Horsens har vi fået følgende recept på, hvordan problemerne kan bringes ud af verden:

1. Linie 26027 fjernes. Efter at programmet er indlæst i computeren, tastes 26027, hvorefter man trykker RETURN.

2. Linie 10020 skal se således ud:

```
GOSUBB 26012 : N = NR + 1
:PRINT LEFT$(N$,19)
+ SP$ + SP$;
```

3. Linie 10030 skal se således ud:

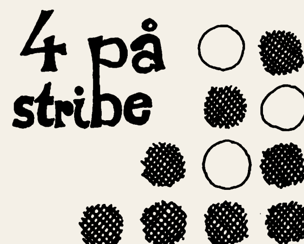
```
GOSUB28000:IFCI$ =
''S''THEN 1000
```

4. Linie 50005 skal se således ud:

```
L% = L% + 1:PRINTC$;:
IFL%    LL%THENL%
= L% - 1:PRINT''        '';:
GOTO50001
```

Læg mærke til, at indholdet mellem de to citationstegn i linie 50005 skal være cursor til venstre, mellemrum og cursor til venstre.
Når man har rettet fejlen, er der bare tilbage at SAVE programmet i den nye og forskønnede version.

VI & VIC-bånd med fire på stribe-spil



I VI & VIC nr. 3 gengav vi et program, som kunne lave VIC om til et fire på stribe-spil.
De har været meget stor interesse for dette spil - men desværre har mange læsere også haft visse vanskeligheder med at få det til at fungere. Det vidner en række henvendelser til redaktionen om. Derfor har vi valgt at tage programmet med på listen over bånd, som kan erhverves for tyve kroner. Se bestillingslisten.
Men programmet, som vi har indspillet på bånd, er ikke helt identisk med det, der var gengivet i bladet. Først og fremmest er grafikken forbedret, så »brikkerne« ikke skal passere tværs igennem de streger, der danner spillepladen. Når en af brikkerne udløses, åbner der sig en lem, brikken kan falde igennem, og de vandrette streger bliver »skubbet« til side. Når brikken er faldet helt ned, lukker stregerne sig igen.

Lethedsgrader

En anden ændring i programmet består i, at brikkerne selv bevæger sig frem og tilbage foroven i skærmbilledet, så man blot behøver at trykke på en tast, når de er lige over den kanal, man har udset sig. Den hastighed, brikkerne skal bevæge sig med, kan man vælge før spillets start. Der spørges om »Hvor let«, og man skal vælge et tal mellem 1 og 9. Jo højere tal, jo lettere spil. Ikke alene er der forskel på, hvor hurtigt, brikkerne kører frem og tilbage foroven, der er også indbygget en »time-out«-funktion, som medfører, at brikken falder ned, hvis man ikke tilstrækkeligt hurtigt har valgt en placering.
Det er sådan indrettet, at denne »time out« ikke findes i den letteste.
Det er ikke kun, når man spiller fire på stribe-spillet, man skal være hurtig. Muligheden for at vælge lethed består også kun et begrænset stykke tid (mellem fire og fem sekunder). Herefter vælger programmet selv en middel sværhed.
at programmet fylder 2406 bytes, så det kan køre i en VIC uden udvidet hukommelse. Det er også skrevet, så det ikke uden ændringer kan køre i en VIC med mere end 8 K RAM i alt.

Programmer er illustrationer til artiklerne

De programmer på bånd, der kan købes fra VI & VIC, må ikke betragtes som andet end illustrationer til de artikler, der har været bragt i bladet.
Først og fremmest må vi bede om forståelse for, at programmerne KAN have skavanker, der ikke ville være acceptable i færdigudviklede programmer, som har været igennem den aflusningsproces, der bl.a. kræver lang tids brug.
Meningen med at tilbyde læserne disse bånd er, at det kan være et stort og trælsomt arbejde at taste længere programmer ind - og vi har erfaret, at mange har vanskeligheder, som man på denne måde let og elegant kan springe hen over. Programmerne kan også betragtes som et udgangspunkt for at lege videre med programmeringen: Lægge andre funktioner ind, forbedre de eksisterende og naturligvis altid peppe billedet op med farve - eller lægge lydeffekter ind i afviklingen af programmet.

Commodore 64

Vi er blevet spurgt, om programmerne kan køre på Commodore 64, og hertil må vi desværre sige, at det kan de ikke. I hvert fald ikke uden en forudgående bearbejdelse. Alene af den grund, at der er én eller anden form for lydeffekter i programmerne, vil det gå galt i en 64'er, fordi her anvendes POKE OG PEEK - og adresserne i VIC-20 og Commodore 64 er ikke ens. I en del af programmerne anvendes også PEEK og POKE til skærm og farver, og disse instruktioner skal i givet fald ændres.

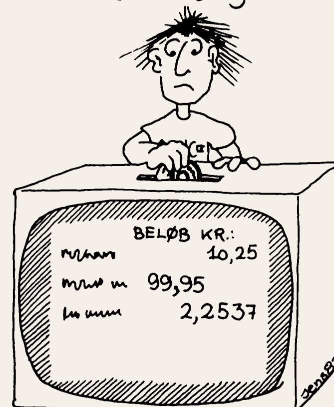
Programlistning

Redaktionen kan desværre ikke påtage sig at foretage disse ændringer til de eksisterende programmer, og vi har heller ikke mulighed for at tillempe programmerne til andre behov end dem, de blev skrevet til. Vi er f.eks. blevet bedt om at lave en version af ØKO SYS, der kan køre med diskettestation, og der har været ønsker om mulighed for at tilslutte printer.
Her er det, vi mener læsernes kreativitet må fortsætte, hvor vores ben-arbejde slap op. Men det skal også straks erkendes, at det kan være vanskeligt at gå i lag med redigering af et vanskeligt overskueligt program. Derfor vil vi nu efterkomme ønsket om at levere en programlistning sammen med båndet - eller man kan nøjes med denne programlistning, hvis man har mod på at tage et par timer ved tastaturet.
Prisen for disse programlistninger har vi sat til 10 kr. inkl. moms og inkl. forsendelse, men vi må bede om, at beløbet sendes sammen med bestillingen på samme måde, som det gælder for bånd og memory maps.

Nu kommer bogen om programmering af VIC-20 og Commodore 64

»Programmering med Commodore Basic« er titlen på en ny bog, som ventes på gaden i slutningen af april måned.
Det er Borgens Forlag, der vil forfølge den succes, man har haft med en tilsvarende bog om programmering af ZX 81-datamaten, og forlagsredaktør Niels Borgen oplyser, at denne bog er solgt i mere end 6000 eksemplarer, så han venter også en stor interesse omkring den nye bog.
Ligesom ZX 81-bogen er »Programmering med Commodore Basic« skrevet af den danske forfatter Erwin Neutzsky-Wulff.
Bogen beskriver den BASIC-version, som findes i VIC-20 og Commodore 64, og Niels Borgen siger, at der indledes med et regulært grundkursus i BASIC-programmering - lige fra at fortælle, hvad der sker, når man trykker på én af computerens knapper til de mere raffinerede muligheder som f.eks. definerbare funktioner, som også findes i BASIC.
I bogens sidste halvdel tages læserne med ind i den mere specielle VIC-verden med en kulegravning af de muligheder, der ligger i VIC-20. Igennem hele bogen krydres lærdommen med programeksempler, så læseren i praksis får mulighed for at afprøve sine færdigheder.
Bogen om programmering af VIC og Commodore 64 bliver på ca. 260 sider og kommer til at koste 140 kr.

En subrutine, der runder kr. og øre af - og sætter beløbene pænt i kolonner




```
700 TR$=TR$+"..............................................................
......."
701 PR=INT(PR*100+.5)/100:PR$=STR$(PR):IFPR<1ANDPR>0THENPR$="0"+MID$(PR$,2):GOTO
704
702 IFPR-INT(PR)=0THENPR$=PR$+"."
703 IFPR<0ANDPR>-1THENPR$="-0"+MID$(PR$,2)
704 IFASC(RIGHT$(PR$,3))<>46THENPR$=PR$+"0":GOTO704
810 HL$=TR$+PR$:IFLEN(HL$)>LLTHENTR$=LEFT$(TR$,LEN(TR$)-1):GOTO810
820 IFLEN(HL$)<LLTHENTR$=TR$+" ":GOTO810
830 PRINTHL$:RETURN

READY.
```

Her er et hjælpemiddel for den, der har brug for at stille beløb op under hinanden. Teksten til beløbet kommer til at stå til venstre, og beløbet rundes af til to decimaler og placeres pænt under hinanden til højre. Det er valgfrit at få en række punktummer til at slutte linien ud og lette læsningen.

For at disse programlinier, der kan anvendes som en subrutine i mange slags programmer, skal tre variable lægges fast. Først og fremmest linielængden, LL. Hvis man opererer med den samme linielængde hele tiden, kan det gøres i begyndelsen af programmet med en instruktion som f.eks.: LL = 20, hvis man udelukkende vil arbejde på skærmen - eller LL = 60, hvis man arbejder med en printer. Det er naturligvis muligt at fastsætte linielængden hver gang, subrutinen anvendes. - På den måde kan indtægter og udgifter få hver sin kolonne.

Dernæst skal teksten defineres i variablen TR$, og beløbet, der skal printes, indsættes i variablen PR.

Sådan virker subrutinen

I linie 701 foretages øre-afrundingen, hvorefter den numeriske variabel PR omdannes til en strengvariabel PR$.

Men allerede her giver VIC os problemer. For hvis tallet er mindre end 1, får vi ikke noget nul foran punktummet (der fungerer som komma). Det løses også i linie 701, hvor PR$ i tilfælde af tal mindre end 1 men større end nul defineres som et NUL efterfulgt af den oprindelige PR$'s 2. og følgende karakterer. På denne måde bortfiltreres den blanke karakter, som indleder de numeriske variabler - og som også overføres til strengbariablen.
have løst, er det, at VIC ikke sætter noget punktum, når tallet ikke har nogen decimaler. Den sag klares i linie 701, hvor der tilføjes et punktum, hvis vort tal er helt uden decimaler.

Tal mellem 0 og minus 1 får i linie 703 anbragt et minus og et nul foran decimal-punktummet.

Er der ingen decimaler - eller kun én, mangler vi de nuller til sidst, som får tallet til at tage sig pænt ud. Det sørger linie 704 for at råde bod på. Her kontrolleres, om det tredie sidste tegn i strengen PR$ har ASCII-koden 46, der er betegnelsen for et punktum. Er det ikke tilfældet, tilføjes et nul og teksten gentages, indtil der er kommet det rigtige antal nuller på vort tal.

Hele linien

Nu har vi langt om længe fået vores tal til at antage et format, som passer til vores ønsker om et beløbs udseende, men vi har endnu ikke fået det placeret rigtigt på skærmen (eller papiret).

Dette arbejde starter i linie 810, hvor en ny variabel HL$ sættes sammen af tekstvariablen TR$ og beløb-variablen PR$. Når denne variabel er nøjagtig en linie lang, kommer kolonnerne til at passe. Først kigges der efter, om den er for lang. Er det tilfældet, skæres den sidste karakter af teksten, og der testes pány i linie 810.

I linie 820 kontrolleres, om linien bliver for kort. Er det tilfældet, forlænges teksten med et ordmellemrum - og der startes forfra med kontrollen af linielængden. Når alt er i orden PRINTes linien HL$ og subrutinen er afsluttet.

Hvis man gerne vil have prikker fra teksten hen til beløbet, starter man blot linie 700, hvor teksten tilføjes et stort antal punktummer (der for de flestes vedkommende bliver fjernet igen i linie 810). Når man ønsker prikker, skriver man altså GOSUB 700. Ønsker man ingen prikker, skriver man GOSUB 701.

Lad VI & VIC hjælpe dig

VI & VIC skal være med til at du kan få mere fornøjelse af din VIC-20, og derfor indbyder vi dig til at komme med dine praktiske eller mere teoretiske spørgsmål.
Pladsen vil naturligvis sætte grænserne for, hvor mange spørgsmål, der kan blive besvaret her i bladet, men vi vil naturligvis først og fremmest lægge vægt på at besvare de spørgsmål, som optager mange VIC-brugere.
Skriv til VI & VIC på adressen Kornager 29, 7100 Vejle, hvis du har spørgsmål, du gerne vil have behandlet.

FORTH - sproget til VIC, som man selv kan udvide og definere, som man vil

Der findes i dag en mængde programmeringssprog, man kan næsten ikke åbne et computerblad uden at se et nyt sprog beskrevet, så hvad skal vi nu med et sprog til?
FORTH er ikke noget nyt sprog, men det er ikke ret kendt ud over de professionelles rækker, og det er egentlig synd, for FORTH er særdeles velegnet til programmering af microprocessorer. De fleste af os har da også brugt FORTH programmer uden at vide det, for utallige af de spil, man ser på cafeterier og på markedspladser, er programmeret i FORTH, ligeså microprocessorstyrede radioer, opvaskemaskiner, køleskabe o.s.v.

Effektivt

Industrien bruger FORTH i stor udstrækning, fordi det er et effektivt sprog, der giver en god udnyttelse af maskinen og samtidg giver en meget kort programudviklingstid.
Lad os prøve at sammenligne FORTH med de to programmeringssprog, hobbyfolk normalt kommer i berøring med, ASSEMBLER og BASIC.
Basic er et fortræffeligt sprog til førstegangs computerbrugere, men skal man lave programmer af en vis størrelse, begynder vanskelighederne at melde sig. Det er ikke nemt at strukturere, og prøver man at skabe sig et bibliotek af subrutiner, for dog ikke at skulle skrive de samme ting om og om igen, får man besvær med at holde rede på sine variable navne, linienumre m.m. Hvad værre er, Basic er en interpreter, d.v.s. de enkelte linier af programmet bliver oversat igen og igen, hver gang de skal bruges, det går ud over hastigheden.

Hastigheden

Skal man bruge sin computer til noget, hvor hastigheden spiller en væsentlig rolle, er man nødt til at bruge Assembler, eller en kombination af Assembler og Basic. I Assembler definerer man hver enkelt maskininstruktion, man ønsker udført, det kræver et betydeligt kendskab til maskinen, og det er vanskeligt at linke Basic og Assembler rutiner sammen via SYS eller USR funktionerne. Det allerværste er, at programmer, der er fulde af PEEK's og POKE's og Assembler rutiner og som er skrevet for en VIC-20, ikke uden et betydeligt arbejde og indgående kendskab til begge maskiner, kan bringes til at køre på f.eks. en PET.
Når nu så mange professionelle ser deres fordel i at bruge FORTH, hvor sådanne problemer slet ikke opstår, så lad os prøve at se nærmere på det.

FORTH sproget er ikke noget programmeringssprog i gængs forstand, det er et bibliotek eller en ordliste af kommandoer eller rutiner, som man kan sammenstille uden at bekymre sig om overførsel af variabler fra den ene rutine til den anden.
Disse FORTH ord, som man kalder dem, gemmes i en ordliste, og FORTH's store fordel består i, at man nemt kan definere, afprøve og derefter

Et bibliotek

anbringe sine egne ord i ordlisten.
Siden alt i FORTH drejer sig om ord, må vi hellere slå fast, hvad et FORTH ord er, alle programmeringssprog har et eller andet regelsæt, som definerer sproget, men man finder næppe simplere regler end FORTH's.

FORTH regel nr. 1:
Den grundlæggende enhed i FORTH kaldes et ord. Et ord består af en række af et eller flere bogstaver, tegn eller tal, som er afgrænset af mellemrum. Eks.: **FORTH + IF */ 2§** er eksempler på fem ord, der opfylder reglerne for et FORTH ord.

Data

Men hvor bliver vores data af? Hvor gemte ordet **PADDLE** adressen på vores port, og hvor gjorde ordet **C @** af vores paddle status? Det bringer os til FORTH regel nr. 2:
Parameter og dataoverførsel mellem FORTH ord sker ved hjælp af en LIFO stack.
LIFO står for last-in-first-out og en LIFO stack kan bedst sammenlignes med en bunke. Man lægger ting på toppen af bunken, som altså bliver højere, og den sidst lagte ting ligger øverst. Når man fjerner noget fra bunken, får man fat i de øverste ting først, og bunken bliver mindre, indtil den er tom.
Lad os prøve at skrive **READ** igen. FORTH interpreteren vil lede i ordlisten og finde definitionen af **READ.** Først udføres **PADDLE**, og **PADDLE** vil anbringe adressen af vores paddle port på stacken. Det næste ord var **C @** (C.fetch), et standard ord der ifølge definitionen vil tage et tal fra stacken, bruge det som storrage adresse og lægge indholdet af denne adresse tilbage på stacken, så møder vi ordet ; der fortæller os, at vi er færdige, d.v.s. efter at vi har udført definitionen for **READ**, er vores stack blevet én position højere, og på toppen ligger vores paddle status, det var faktisk også det, vi ønskede.
Ord som ikke findes i ordlisten, men som kan konverteres til tal, bliver også anbragt på stacken, hvis vi skriver:
2 4 6 8 (return) vil vi have en stack med fire tal og 8, som jo blev skrevet sidst, vil ligge øverst. Ordet . (dot) er et standard ord, som vil tage det øverste tal på stacken og skrive det på vores output device. Så hvis vi nu skriver fem punktummer (dots), husk mellemrum mellem hver, da det jo er FORTH ord, vil vi se følgende:
. . . . . 8 6 4 2 0 .
STACK EMPTY!
Vi ser, at tallene bliver skrevet ud ét af gangen, men vi puttede kun fire tal ind, og vi prøvede at skrive fem tal ud. FORTH vil skrive et eller andet tilfældigt i stedet for det femte tal og derefter give en fejlmeddelelse: **. STACK EMPTY!** d.v.s. ordet **.** prøvede at fjerne noget fra en tom stack.

FORTH gemmer sine ord sammen med en difinition af det arbejde, man vil have udført, når man bruger det pågældende ord i sine ordlister, man kan selv bestemme, hvor mange ordlister man vil have, men man skal bruge mindst én.
Når FORTH's interpreter rutine møder et ord i sin inputlinie, vil den lokalisere ordets definition i ordlisten ved hjælp af ordets navn. Hvis ordet findes, udføres definitionen. Hvis ordet ikke kan findes i ordlisten, vil interpreteren antage, at det er et tal og prøve at konvertere til den interne kode for tal, normalt seksten bit heltal. Skulle

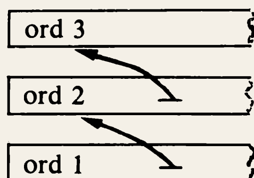
ordet indeholde tegn, som ikke kan konverteres, vil FORTH udskrive en fejlmeddelelse.

Nye ord

Nye ord adderes til ordlisten ved at bruge et specielt ord : (colon). Lad os tage et eksempel:
Hvis vi tidligere har defineret et ord **PADDLE**, som giver os adressen på vores paddle eller joystik, og vi ønsker at definere et nyt ord **READ**, som skal læse input fra vores paddle port, skriver vi:
: READ PADDLE C @;
Colon er det ord, der fortæller, at et nyt FORTH ord skal adderes til ordlisten, nemlig **READ**, og i definitionen af **READ** indsættes henvisninger til **PADDLE, C @** og ;.
PADDLE havde vi jo selv defineret tidligere, C @ (C-fetch) er et ord, der vil læse én byte fra en storrage adresse, og ; (semi-colon) er et ord, der fortæller interpreteren, at orddefinitionen skal afsluttes. Nu er ordet **READ** kendt af FORTH, så når vi i fremtidige definitioner skriver **READ**, vil FORTH give os status af vores paddle port.

Retur stacken

Lad os tage FORTH regel nr. 3:
En fejlmeddelelse standser eksekveringen og tømmer begge stackene. Vi har kun omtalt parameterstacken, men der findes en stack mere kaldet retur stacken.
Når vi møder et ord, der refererer til et andet ord, der igen refererer til et tredie ord



siger vi, at vi går et level ned, vi bruger så retur stacken til at gemme adressen på stedet, vi kom fra, således at vi kan finde tilbage igen og fortsætte det rigtige sted, når vi møder et ;, men det vil vi komme nærmere ind på i en senere artikel om de indre mekanismer i FORTH.
Kan vi nu med så simple midler som en interpreter to stacks og en ordliste løse problemer, som man ellers skal bruge komplicerede kompilere eller interpretere til? Svaret er ja, hvis ordlisten indeholder de rigtige ord.

100-150 ord

Når man anskaffer sig et FORTH system, vil ordlisten indeholde 100-150 omhyggeligt udvalgte ord, som kan bruges til at manipulere stacken med, til at lave beregninger med, og til at lave pænt formaterede udskrifter med. Det er ikke noget problem at udskrive tal i pæne kolonner med FORTH, de eneste ord, man selv skal definere, er dem der hører til de opgaver, man ønsker løst.

Nu er det jo ikke særlig praktisk, at skulle definere sine ord ét af gangen. Derfor gemmer man sine orddefinitioner på eksternt medie i 1 K blokke eller SCREENS, som man kalder dem FORTH, og FORTH systemet indeholder en text editor, hvormed man kan generere og opdatere disse SCREENS.
Foruden text editoren indeholder FORTH systemet også en assembler, som man ved hjælp af ordet CODE kan bruge, hvis man ønsker at definere nogle ord i maskinsprog i stedet for højniveau ord.
Hvis man taster de ca. 100 SCREENS ind, et FORTH system består af, kan man generere et komplet nyt system. Det vil fylde ca. 8K, og man kan lave store programmer på et 2K arbejdslager, for et FORTH program fylder faktisk mindre i storrage end den tilsvarende opgave skrevet i assembler, men det vil vi komme tilbage til, når vi skal »kigge ind i« FORTH.

Teletekst er ikke en sag for VIC

Kære VI & VIC.
Først tak for de dejlige blade, jeg fik efter indmeldelsen, og så til et par spørgsmål: 1. Er der nogen mulighed for at få VIC-20 til at opfange og senere vise de signaler på hvilke Danmark og Sveriges Radio udsender teletekst?
2. Kan man ved at tilføre 16K-RAM modulet, en given spænding fra et element, få den til at huske ikke at glemme, når man slukker for VIC'en, hvis 1 eller 2 kan besvares med ja then goto: omgående opgivelse af på hvilken måde og med hvilke dimser dette udføres.
Til slut vil jeg bede jer overbringe min dybfølte tak til de mennesker, der svarer på mine dumme spørgsmål, når jeg frækt tillader mig at ringe til Commodore både her og over there, magen til venlig og behagelig behandling får man ikke mange steder, og - selv om disse mennesker sikkert utallige gange har svaret og vejledt andre af min endnu famlende, fumlende og fejlfyldte person - ja, så fornemmer man varmen og interesse strømme ud af telefonen.
Så nu har jeg ikke tid til at skrive mere, nu er der gået flere minutter siden jeg sad hos onkel VIC, så nu får jeg abstinenser, tak for nu, tak for optagelsen, tak for, forhåbentlig hurtigt og positivt svar.

Med venlig hilsen
Henning Rugaard

Teletekst er næppe en sag for VIC-20. Vi har i hvert fald ikke noget bud på, hvordan det skulle kunne løses - men med hensyn til at sætte batterier til RAM-moduler, kan vi give et helt klart svar: Den går ikke. Hvis det er et problem, at hukommelsen slettes, når computeren slukkes, har vi imidlertid en meget enkel løsning på dette problem: Lad være med at slukke for den - og hvis det er strømafbrydelser, der er problemer, vil det nok være rimeligere at bygge en nødstrømsforsyning, der kan holde liv i programmer og data.

Men anstrengelserne skal jo ikke være ret store, før der er andre og mere nærliggende måder at sikre sig - f.eks. ved helt enkelt at lagre programmer og data på bånd eller diskette.
PS: Vi har vist brevet til Commodores system-afdeling. Man rødmede.

VIC anvendes til en mere effektiv sprogundervisning

Mens skolelærere strides om indførelsen af edb i undervisningen, sker der lige så stille tiltag i den kommercielle undervisning. Ihvertfald har sprogskolen International Business Languages i København udviklet sprogkurser og tests på datamat. - Og den datamat, der blev udset til opgaven, er VIC-20.

I første omgang bruges sprogkurserne på skolen, men det er meningen, at med tiden skal eleven kunne få kassettebåndene med hjem til sin egen hjemmedatamat.

Teknikere

Skolens leder, Benny Larsen, oplyser, at når netop denne skoles undervisning vil basere sig på hjemmedatamater, hænger det sammen med, at størstedelen af skolens elever er teknikere, ingeniører eller folk fra edb-branchen. Skolen har 34 faste lærere og omkring 5000 elever om året.

Ideen om de datamatbaserede sprogkurser blev sat i værk for halvandet år siden i samarbejde med Commodore, der har ståët for programmeringen. Der anvendes VIC-20 i udvidet version med 16K. Det enkelte bånd, det enkelte kursus indeholder seks emnevalg med tre delemner med hver 30 spørgsmål, således at et kursus ialt indeholder 540 spørgsmål. De fordeler sig på flere niveauer og typer. Lige fra begynderniveau og op til en relativt høj sværhedsgrad og kan bruges dels i en normal øvelsessituation og dels som test. Alt er programmeret i Basic.



Benny Larsen (forrest) og Claude AA Allouche har haft støtte fra Commodore til at udvikle de BASIC-programmer, der bruges i sprogundervisningen, dels som præ-test, dels som egentlig træning. (Foto: Jacob Maarbjerg).

Ud af busken

Benny Larsen understreger, at det langt fra er meningen, at den datamatunderstøttede undervisning skal erstatte lærerne. Det er helt udelukket, og datamaten skal rundt regnet kun bruges i ti procent af undervisningstiden. - Men netop for vort klientel har det vist sig hensigtsmæssigt og meget motiverende. Mange bruger en datamat i deres daglige arbejde, og derfor har vi på denne måde fået et hjælpemiddel til at få folk ud af busken.

På nuværende tidspunkt er kun engelskkurset færdigt, men de resterende hovedsprog er på bedding. IBL har ikke investeret voldsomt i projektet, kun omkring 50-60.000 kr. Udover muligheden for at tage kassettebånd med hjem arbejdes der også på at samkøre programbånd og gængse lydbånd og desuden på programstyret videoundervisning, således at spørgsmål-svar situationen klippes ind i en videofilm. Digitaliseret sprogundervisning afviser IBL på forhånd. Når investeringsudgifterne ikke løber op i svimlende summer, hænger det også sammen med en vis goodwill fra Commodores side. Her har man lige fra starten været dybt interesseret i projektet og kastet en del programmeringsressourcer i sagen.

Samme program kan køre på to måder ved hjælp af WAIT 653,1

Det er almindeligt, at programmer bringes til standsning med en GET-kommando, der f. eks. kan se således ud:

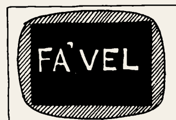
100 GET S$:IFS$ = " " THEN 100

Programmet bliver stående i linie 100 og venter på, at man trykker på en tast - lige meget hvilken, hvorefter det fortsætter.

Med WAIT kan man opnå samme fordel - og man kan endda udnytte en bestemt kommando til at vælge mellem to forskellige måder at afvikle det samme program.

Det drejer sig om WAIT 653,1 - eller den omvendte WAIT 653,1,1. WAIT-kommandoen virker på den måde, at programmet standser og først fortsætter, når indholdet af den specificerede hukommelsesplads svarer til tallet efter kommaet.









SHIFT eller ej

I dette tilfælde arbejder vi med hukommelsesplads nummer 653, som er det sted, VIC gemmer oplysningen om, hvorvidt SHIFT-tasten er trykket ned eller ej. For at se, hvordan det virker, kan man lave et lille program, der ser således ud:

10 PRINT PEEK (653):GOTO 10

Når dette miniature-program køres, vil skærmen få en hel række nuller PRINTet, men så snart SHIFT-tasten trykkes ned, PRINTes der et-taller i stedet. Slip SHIFT, og vi får igen nuller på skærmen.

Her har vi nøglen til en nem måde at gribe ind i et program -uden at standse det. Ønsker man f. eks. at vise mellemresultater af en beregning på skærmen, indlægger man en WAIT 653,1. Bruger man denne metode, kan man også arbejde med WAIT 653,1,1, der standser programmet, indtil SHIFT-tasten slippes.

Betingede hop

En anden mulighed for at benytte kendskabet til hukommelsesplads 653 ligger i at lægge betingede hopinstruktioner ind i programmet, hvor betingelsen netop drejer sig om indholdet af hukommelsesplads 653.

F. eks. kan man skrive:

100 IF PEEK (653) = 1 THEN 200

Denne linie vil få programmet til at springe linierne 101 til 199 over - men kun, hvis SHIFT-knappen er trykket ned. På tilsvarende måde kan man ændre hastigheden på et program ved at benytte indholdet af 653 i en forsinkelsesløkke. Den kan f. eks. komme til at se sådan ud:

FOR I = 1 TO 1000 + 2000* PEEK (653):NEXT

Med SHIFT nedtrykket vil forsinkelsen være omkring tre gange så lang, som når SHIFT ikke er trykket ned.

VIC-brugere i alle byer! Foren jer

Glæden ved at arbejde (eller lege) med VIC-20 kan ofte forhøjes ved at gøre det sammen med andre - men problemer kan undertiden være at finde frem til ligesindede der, hvor man bor.

Derfor vil VI & VIC tilbyde sig som bindeled mellem VIC-brugere, som ønsker kontakt, hvad enten det drejer sig om at hjælpe hinanden med programmering eller andre tekniske problemer, eller det bare gælder kammeratligt samvær.

Benyt kuponen her på siden, hvis du gerne vil have dit navn og adresse i bladet. Det koster ingenting, men til gengæld må meddelelsen være kort - og den må ikke have et kommercielt indhold. Men der må gerne stå et par ord om det, hvis der er specielle problemer, der interesserer dig i VIC-sammenhæng.

På samme måde vil VI & VIC også være behjælpelig med at støtte lokale klubarrangementer ved at viderebringe meddelelser i bladet om tid, sted og emne for møderne. Men husk, at VI & VIC kun er på gaden hveranden måned, så det gælder om at være i god tid. Redaktionen slutter tre uger før udgivelsen, som er den første dag i alle de »lige« måneder.

Hvis du ikke vil klippe i bladet, er det i orden at skrive kuponen af eller tage en fotokopi.

VIC-kontakten

Navn: ____________

Adresse: ____________

____

Postnr.: ____________

BY: ____________

Tlf.: ____________

Bemærkninger: ____________

____

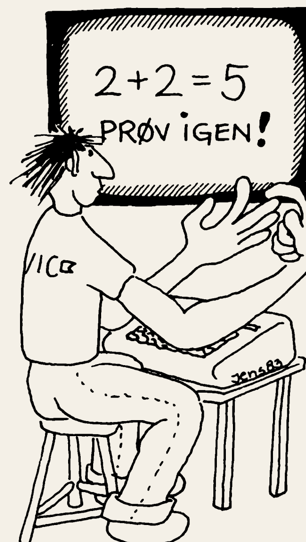
____

____

____

Kuponen sendes til VI & VIC, Kornager 29, 7100 Vejle.

Hjælp til lektierne naturligvis med VIC



De regnelærer-programmer, der har vundet en vis udbredelse, er træningsprogrammer, hvor computeren formulerer en opgave og giver eleven mulighed for at prøve kræfter med den. Derefter kontrollerer programmet, om der er regnet rigtigt og giver eventuelt en chance til.

Men hvad gør man, hvis eleven selv vil skrive sin opgave (fra regnebogen)? Programmet skal gerne køre let og ubesværet, så computeren ikke bare kommer til at fungere som en lommeregner, hvor man bagefter regner ud, om man nåede til det rigtige resultat. Denne opgave har givet mindst én af vore læsere grå hår i hovedet, for problemet kræver, at man går en lille omvej. Det gælder om at få lagt et mere eller mindre sammensat udtryk ned i en variabel i computeren - mens programmet afvikles.

To løsninger

Vi kan her tilbyde to løsninger. En nem og en lidt mere kompliceret, men også mere avanceret løsning. En af vanskelighederne består i at få computeren til at forstå de parenteser og regnetegn, som ikke kan bruges i en INPUT-sætning. Man kan selvfølgelig lade computeren læse udtrykket som en streng-variabel, men det får man ikke udført noget regnestykke af.

I den korte løsning skriver computeren på skærmen:
OPGAVE =
og derefter bruges GET$ til at hente tegnene fra tastaturet. I vort lille eksempel er den blinkende cursor på skærmen hjemmelavet (i linie 30). Efterhånden som man skriver sine tegn, bliver de PRINTet på skærmen, og når man er færdig med at definere sit regnestykke, trykker man RETURN.

Programmet slutter

Herved standser programmet, og på skærmen står READY, og nedenunder blinker cursoren - men ikke ret længe. For lige inden END-kommandoen i linie 199 er der sket en hel masse spændende ting i linie 100 og 110.

Her er nemlig på skærmen blevet PRINTet »GOTO 200«, og derefter er der POKEt en række tal ind i de hukommelsespladser, der hører til tastaturets buffer. D.v.s. det lille lager, som gør det muligt at skrive en lille smule foran computeren.

I linie 110 står der først POKE 198,8. Denne ordre giver VIC besked om, at der står 8 tegn i tastaturbufferen og venter.

Derefter POKEs tallet 145 ind i pladserne 631 til 636 - hvilket får cursoren til at bevæge sig seks linier op på skærmen, når den første del af programmet slutter. De to sidste pladser i tastaturbufferen, som vi har udnyttet er 637 og 638, hvor vi har gemt tallet 13. 13 står for RETURN, og den første RE-

```
REGNESTYKKER
************


10 PRINT"▒▒▒SKRIV OPGAVEN▒":PRINT"OG TRYK ▒RETURN▒"
20 PRINT"▒OPGAVE= ";
30 GETA$:PRINT" ▮";:FORT=1TO100:NEXT:IFA$=""THENPRINT"▒▒▒";:FORT=1TO100:NEXT:GOT
O30
35 IFA$=CHR$(13)THEN100
36 IFA$=CHR$(20)THEN60
40 IFA$<"("ORA$>"9"THEN30
50 IFA$=","THEN30
60 PRINTA$;:GOTO30
100 PRINT:PRINT:PRINT:PRINT"G⌐200"
110 POKE198,8:FORT=631TO636:POKET,145:NEXT:FORT=637TO638:POKET,13:NEXT
199 END
200 PRINT"▒▒▒":FORT=1TO3:PRINT"          ":NEXT:PRINT"HAR DU EN LØSNING?"
210 INPUTL:IFL=OPTHENPRINT"DET ER HELT RIGTIGT":GOSUB999:RUN
220 PRINT"FORKERT":F=F+1
230 IFF<3THENPRINT"PRØV IGEN":GOTO210
240 PRINT"LØSNINGEN ER":PRINTOP:GOSUB999:RUN
999 GETS$:IFS$=""THEN999
1000 RETURN
```

TURN udløses, når cursoren befinder sig i det regnestykke, vi skrev på skærmen. Resultatet bliver derved lagt ned i variablen OP(gave). På dette tidspunkt befinder cursoren sig på linien med ordene GOTO 200, og her udløses den anden RETURN, hvilket får programmet til at starte igen - men denne gang fra linie 200. Det er nødvendigt at bruge GOTO og ikke RUN, fordi RUN ville slette alle de variable, og altså også resultatet af vort regnestykke.

På denne måde har vi opnået, at computeren kender resultatet af regnestykket, uden at det har stået på skærmen, og sidste halvdel af programmet giver nu brugeren tre chancer til at regne rigtigt. Vi vil overlade det til interesserede læsere at peppe programmet op med elegante skærmbilleder og passende lyd til at indikere rigtigt eller forkert svar.
Læg mærke til, at der i begyndelsen af programmet er taget forholdsregler imod at andet end tal, regnetegn og parenteser accepteres.

Evaluator

Det program, vi netop har gennemgået, fylder ikke meget, men det har den ulempe, at det kræver skærmen til sin rådighed, og det kan måske være et problem.
Vi ser heller ikke bort fra den mulighed, at der er læsere, som foretrækker en mere sofistikeret løsning, og den følger så her i form af en »Numerisk evaluator«.
En meget central linie i programmet er linie 10, hvor variablen Z er defineret som en masse tomme pladser efterfulgt af + 0.
Nu gælder det bare om at få anbragt de tal, parenteser og regnetegn, der er brug for i linie 10 efter lighedstegnet og før + 0. Det sker ved at programmet undersøger de tegn, man taster ind for at sikre, at de er acceptable, hvorefter de POKEs ned i brugerRAMMEN -faktisk lige nøjagtig ned i programmets linie 10. Denne manøvre vil kunne lykkes, hvis man har nøjagtig styr på, **hvor** i RAM, de tomme pladser i linie 10 befinder sig - og skriver man programmet **nøjagtig** som det er gengivet her, vil de tomme pladser være at finde fra hukommelsesplads 4218 som angivet ved programmets start (RA = 4218). Har man udvidet hukommelsen til mere End 8 K i alt, skal dette tal forøges med 512, så der blot skal skrives RA = 4730 i linie 1.
I begge programmerne kan man anvende DELete-knappen.

```
EVALUATOR PROGRAM
*****************


1 RA=4218:GOTO200
2 Z1=Z1+1:POKE(RA+Z1),ASC(Z$):IFZ1=70THEN14
4 GOTO35
6 IFZ7<>0THEN14
8 IFZ1<70THENFORZ2=Z1+1TO70:POKE(RA+Z2),32:NEXTZ2
10 Z=                                                                      +0
12 IFZ$=CHR$(36)GOTO14
13 PRINT:RETURN
14 PRINT"RIP"
30 Z1=0:Z2=0:Z6=0:Z7=0:Z8=0:Z9=0:Z$=""
35 GETZ$:IFZ$=""THENGOSUB142:GOTO35
40 IFZ$<>CHR$(20)THEN55
45 IFZ1=0THENZ8=0:Z6=0:GOTO35
50 PRINT"█";:Z1=Z1-1:Z$=CHR$(PEEK(RA+Z1)):GOSUB152:Z8=Z9:Z6=0:GOTO35
55 IFZ$=CHR$(13)ORZ$=CHR$(36)THEN6
60 PRINTZ$;:GOSUB152:ONZ9GOTO104,110,66,68,70,72,74,76,78,80,150
66 Z6=0:Z$=CHR$(170):GOTO110
68 Z6=0:Z$=CHR$(171):GOTO110
70 Z6=0:Z7=Z7+1:GOTO110
72 Z6=0:Z7=Z7-1:GOTO110
74 Z6=0:Z$=CHR$(172):GOTO110
76 Z6=0:Z$=CHR$(173):GOTO110
78 Z6=0:Z$=CHR$(174):GOTO110
80 Z$=CHR$(255):GOTO110
104 IFZ6=0THENZ6=1:GOTO110
106 GOTO150
110 IFZ1=0THENONZ9GOTO140,140,140,140,140,150,150,150,150,140
112 ONZ8GOTO116,118,120,122,124,126,128,130,132,134
116 ONZ9GOTO150,140,150,150,150,150,150,150,150,150
118 ONZ9GOTO140,140,140,140,150,140,140,140,140,150
120 ONZ9GOTO140,140,150,150,140,150,150,150,150,140
122 ONZ9GOTO140,140,150,150,140,150,150,150,150,140
124 ONZ9GOTO140,140,140,140,140,150,150,150,150,140
126 ONZ9GOTO150,150,140,140,150,140,140,140,140,150
128 ONZ9GOTO140,140,140,140,140,150,150,150,150,140
130 ONZ9GOTO140,140,140,140,140,150,150,150,150,140
132 ONZ9GOTO140,140,140,150,140,150,150,150,150,140
134 ONZ9GOTO150,150,140,140,150,150,140,140,140,150
140 Z8=Z9:GOTO2
142 FORZ4=1TO100:NEXT:PRINT"_█";:FORZ4=1TO100:NEXT:PRINT" █";:RETURN
144 GETZ$:IFZ$=""THENGOSUB142:GOTO144
146 RETURN
150 PRINT"█";:GOTO35
152 IFZ$=CHR$(46)THENZ9=1:GOTO174
154 IFZ$=CHR$(43)ORZ$=CHR$(170)THENZ9=3:GOTO174
156 IFZ$=CHR$(45)ORZ$=CHR$(171)THENZ9=4:GOTO174
158 IFZ$=CHR$(40)THENZ9=5:GOTO174
160 IFZ$=CHR$(41)THENZ9=6:GOTO174
162 IFZ$=CHR$(42)ORZ$=CHR$(172)THENZ9=7:GOTO174
164 IFZ$=CHR$(47)ORZ$=CHR$(173)THENZ9=8:GOTO174
166 IFZ$=CHR$(94)ORZ$=CHR$(174)THENZ9=9:GOTO174
168 IFZ$=CHR$(222)ORZ$=CHR$(255)THENZ9=10:GOTO174
170 IFZ$<CHR$(48)ORZ$>CHR$(57)THENZ9=11:GOTO174
172 Z9=2
174 RETURN
200 REM ++++++++ EXAMPLE +++++++++
210 PRINT"] DATA ";:GOSUB30:PRINTZ
```

Lysavis giver instruktioner uden at ødelægge skærmbilledet

Har man brug for at give en længere instruktion på skærmen uden at ødelægge mere end den øverste linie i skærmbilledet, kan dette program bruges - men man kan naturligvis også bruge det uden at det er strengt nødvendigt. Måske er det bare sjovt at lave sin egen lysavis på VIC.

Teknikken i programmet er den enkle, at den tekst, der skal vises i lysavisen, gemmes i data-sætninger i programmer - men der ville naturligvis ikke være noget i vejen for, at man hentede teksten fra bånd eller diskette.

Programmet er konstrueret som en subrutine, der i øverste linie på skærmen PRINTer en streng, der hele tiden gøres et tegn kortere i venstre side, mens der føjes et nyt tegn til i højre side. De nye tegn hentes med READ A$ (i linie 30) hver gang det hidtidige indhold af A$ er »brugt op«. Når programmet har kørt så længe, at man når stjernen, der er den sidste variabel i data-sætningerne, starter man forfra med RESTORE.

For at komme ud af subrutinen, skal man trykke på en hvilken som helst tast. Det vil i linie 32 føre frem til RETURN.

Hvis man bruger denne subrutine i et program, hvor der også på anden måde benyttes READ-instruktioner, må man være opmærksom på risikoen for at få blandet de forskellige sæt data sammen. Den enkleste måde er at benytte RESTORE og så »tælle« sig så langt frem i datasætningerne, der er brug for. I det viste eksempel er der 10 variable gemt i data-sætningerne fra linie 1000 til linie 1010. Hvis man placerer flere data, som skal bruges i en anden sammenhæng, ville man være nødt til at læse de første 10, inden man kan læse dem, det egentlig drejer sig om.



Programmet skulle så indeholde disse linier:
100 RESTORE
110 FOR I = 1 TO 10:
READA$:NEXT
120 READ A$
o.s.v.

```
READY.

10 RESTORE:N$="                              "
20 READA$
30 IFA$=""THENREADA$:A$="  "+A$
31 N$=N$+LEFT$(A$,1):A$=MID$(A$,2):PRINT"■"N$
32 GETS$:IFS$<>""THENRETURN
35 FORD=1TO77:NEXT
40 IFLEN(N$)>21THENN$=MID$(N$,2)
45 IFA$<>"*"THEN30
50 RESTORE
60 GOTO30
1000 DATADETTE ER ET EKSEMPEL DER SKAL VISE HVORDAN MAN KAN FAA
1001 DATAEN LILLE LYSAVIS TIL AT LØBE HEN OVER TOPPEN AF
1002 DATATV-BILLEDET.       DET KAN F.EKS. ANVENDES HVOR MAN VIL
1003 DATA GIVE EN LANG FORKLARING ELLER INSTRUKTION UDEN AT FJERNE
1004 DATADET BILLEDE DER FOR TIDEN ER SYNLIGT.  DET GAAR KUN UD
1005 DATAOVER DEN ØVERSTE LINIE PAA SKAERMEN.      PROGRAMMET ER
1006 DATAINDRETTET SAA DET BLIVER VED MED AT GENTAGE TEKSTEN INDTIL
1007 DATAMAN TRYKKER PAA EN ELLER ANDEN TAST - LIGEMEGET HVILKEN
1010 DATA"                         ",*

READY.
```

Memory map til Commodore 64

Den danske brugervejledning til Commodore 64 nåede at blive færdig (næsten) samtidig med, at denne storebror til VIC-20 blev præsenteret for det danske publikum.

Den er en lille håndbog i A5-format med en praktisk spiralryg, der gør den egnet til at blive brugt - og til at blive brugt meget.

Ligesom i VIC-20-brugervejledningen gives der nye computerbrugere en grundig indføring i BASIC-programmering, så man ikke behøver være bange for at gå i gang med 64'eren. Foruden de for VIC-brugere kendte emner vedrørende anvendelse af farver og grafik er der i brugervejledningen gjort omhyggeligt rede for den mere avancerede brug af farver, der er mulighed for i Commodore 64. Denne maskine råder jo over 16 farver på tegnene, hvor VIC »kun« har otte forskellige.

På samme måde har både sprite-grafikken og den meget avancerede lyd fået hver sit kapitel i brugervejledningen, og man må complimentere forfatterne til denne vejledning med den idé, de har fået med at anbringe Commodore 64 memory map på de sidste sider i brugervejledningen. Mange har brug for mere detaljerede oplysninger om, hvor i computeren, der foregår de forskellige ting - både når man vil bruge PEEK og POKE, og når der er tale om egentlig maskinkodeprogrammering.

VIC programmer

Lad andre få mulighed for at købe de programmer, du udvikler til VIC.

Rubrikannoncerne her i bladet koster 1 kr. pr. spaltemillimeter, og for overskuelighedens skyld bør annoncerne bygges op på samme måde, som det kan ses her på siden. Det vil sige, at annonceteksten skal indeholde oplysninger om nødvendigt udstyr, hvilket medium, programmet findes på (kassettebånd, disk eller cartridge), prisen - og naturligvis oplysninger om, hvor man kan købe programmet.

Brug kuponen til din annonce.

VIC program sælges

Programmets navn: ______________________

Beskrivelse: ______________________

____

____

____

____

____

____

____

____

____

____

____

____

Hardware: ____________ Medium: ____________

Pris: ______________________

Købes hos: ______________________

____

Kuponen sendes til VI & VIC, Kornager29, 7100 Vejle.

Grafik-Editor

Superavanceret editor-program, med masser af hjælpefunktioner, til at lave bruger definerede tegn med. Der medfølger anvisninger om hvordan man gør sine spil mere »Livagtige«, demo-programmer, der viser eksempler på bevægelig grafik samt udførlig anvisning i brugen af editoren.

Grafik-editoren fås også i en mindre avanceret udgave til VIC 20+ 3 K ram (75 kr.).

Yderligere oplysninger kan tilsendes.

Hardware: VIC-20 + 8 K ram.
Medium: Kassette.
Pris: 100 kr. (Kun Check/efterkrav)
Købes hos: Arno Greve Andersen, Vibækvej 15, 5250 Odense SV.

Spil m.m.

Enarmet tyveknægt, Landevejs race, Bio-rytmer med 4 kurver, renteberegning-(Pantebreve), ostespiseren og terningespil for 4.

Hardware: VIC-20 3,5 K ram.
Medium: Kassettebånd.
Pris: 100 kr. med bånd (Check).
Købes hos: Tommy Larsen, Vanløse Alle 2th., 2720 Vanløse, tlf. 01 - 74 60 21.

Monitor

Simpel maskinkodemonitor med kommandoerne .m, .l, .s, .r, .g og .x, samt basic udvidelse med følgende hjælpeinstrukser: find, change, number og delete. Skriftlig brugsanvisning medfølger.

Hardware: VIC, alle størrelser.
Medium: Kassettebånd.
Pris: 120,- kr. (check).
Købes hos: Chr. Dirksen, Anneksgårdsvej 8, Gundsømagle, 4000 Roskilde.

VIC-20

Hjemmecomputeren der vokser med opgaverne



1 VIC-20 computer
Det private computersystem starter med VIC-20 - hjemmecomputeren fra Commodore. Verdens første fuldt udbyggede farvecomputer til **kr. 2.995,-**

VIC-20 kan udbygges til et enkelt eller mere avanceret system efter ønske og i takt med brugerens behov. Til privat brug. Til undervisning. Til forretningsbrug.

VIC-20 har skrivemaskinetastatur. 8 funktionstaster. 255 farvekombinationer. 64 grafiske tegn. Musikgenerator. Intern hukommelse op til 32K RAM. Tilsluttes alle TV-apparater og monitorer.

2 VIC-1530 kassettestation
Det første skridt mange tager, når de udbygger deres VIC-computersystem. Kassettestationen kan bruges til datalager eller til overførsel af programmer. **Pris: kr. 1.035,-.**

3 VIC-1540 diskettestation
Diskettestationen giver mulighed for det fulde udbytte af VIC-computersystemet. Lagerkapaciteten udvides til 170K bytes. Op til 4 diskettestationer kan kobles til samtidigt. Velegnet til programudvikling. Hurtig dataoverførsel. **Pris: kr. 6.495,-.**

4 VIC-1515 printer
Rigtig papirprinter. 80 karakterer og en hastighed på 30 tegn/sek. **Pris: kr. 4.945,-.**

5 VIC-1020 moder-modul
Dette modul er bl.a. nødvendigt for at udnytte VIC fuldt ud. Gør det muligt samtidig at tilslutte f.eks. 3 - 8 og 16K RAM, superexpander, etc. **Pris: kr. 2.280,-.**

6 Udvidelse af hukommelse
VIC-20's hukommelseskapacitet kan udvides med 3K til 27K RAM.

7 VICMON
Gør det muligt at programmere i maskinsprog. **Pris: kr. 400,-.**

8 VIC-1211A Superexpander
Giver højopløst grafik og ekstra kommandoer til plotning, farvelægning, etc. **Pris: kr. 595,-.**

9 VIC-1212 Programmers aid pack
Et godt hjælpeværktøj ved programmering. Giver 14 ekstra kommandoer. **Pris: kr. 495,-.**

10 VIC-1311 joy stick
Til spil og lignende. **Pris: kr. 95,-.**

11 Brugerhåndbog
Der findes let forståelig dansk brugerhåndbog til VIC, som også på enkel og klar måde fortæller om BASIC. Hveranden måned udkommer brugerbladet VI & VIC, som kan købes i løssalg eller abonnement med VIC-nyheder.

Find nærmeste autoriserede VIC-forhandler på listen og kom ind og prøv VIC hjemmecomputeren.

Alle priser er incl. moms.

commodore COMPUTER

05 - 64 11 55 eller 01 - 88 15 05

ØERNE: Ballerup: Mic Kasseapparater, 02-655111. **Bornholm:** Krogh Hansens Radio & TV, 03-998127. **Herlev:** W. Rolf Pedersen, 02-915511. **Hillerød:** Bo-el Data, 02-253131. **Holbæk:** Papirgården, 03-433003. **København:** BN Elektronic, 01-811900 og 01-184555. CR-Commander Radio, 01-343422. L. A. Elektronik Kbh., 01-551540. Betafon Radio, 01-310273. Samfundslitteratur, 01-351942. Poly Data Microcenter, 01-420705. W. Rolf Pedersen, 01-137056. **Lyngby:** BN Elektronik, 02-881900. Polyteknisk Boghandel, 02-881488. **Maribo:** JD Totalinformation, 03-881700. **Odense:** Dansk Totalinformation, 09-126488. EDB Butikken, 09-147660. Fl. Kjerulff, 09-135480. **Otterup:** Werners Radio Elektronik, 09-823333. **Ringsted:** Ahrent Flensborg, 03-610011. **Rødovre:** Datacare, 01-705858. **Slagelse:** Vest Sjællands Mikrodata, 03-534707. **St. Heddinge:** JE Kontorcenter, 03-703332. **Svendborg:** Jens Bach Kontordata, 09-212788. **Viby Sj.:** Baagøe Radio, 02-394100. **Vordingborg:** Kontorforsyningen, 03-770170. **GRØNLAND: Godthåb:** Godthåb Elektronikservice, 21865. **Julianehåb:** Musikhuset Julianehåb, 01-438133. **JYLLAND: Aalborg:** Knud Engsig Kontorudstyr, 08-126666. H. C. Elektronic, 08-142314. **Esbjerg:** Kurt Øland, 05-125299. **Fredericia:** Semicap Data, 05-931565. **Gram:** Teknik Huset, 04-822711. **Grenå:** Djursland Data, 06-326465. **Haderslev:** H. C. Reimers, 04-523714. **Hadsten:** Hadsten Elektronik, 06-980408. **Herning:** Logic Design, 07-221300. **Hjørring:** Norad, 08-960188. **Holstebro:** Pe-Co Computer System, 07-414646. **Horsens:** Dansk Computer Center, 05-611110. **Kolding:** Hauge Rasmussen Foto, 05-520070. **Randers:** Djursland Data, 06-434200. **Ringkøbing:** LP Musik, 07-322511. **Skanderborg:** Bildata, 06-525344. **Skive:** Elektroniklageret, 07-526177. **Struer:** Helmholt Elektronik, 07-852611. **Sønderborg:** G. P. Kontorsystemer, 04-424546. **Vejle:** 2R Data, 05-831600. **Århus:** Akademisk Boghandel, 06-128844. Clemens Papirteknik, 06-133922. Editor, 06-127720.

Hedelund Marketing, Vejle